计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202330746
61(2): 338−350, 2024

ChatModeler: 基于大语言模型的人机协作迭代式需求获取和建模方法

靳东明[1,2] 金 芝[1,2] 陈小红[3] 王春晖[4]
[1](北京大学计算机学院 北京 100871)
[2](高可信软件技术教育部重点实验室(北京大学) 北京 100871)
[3](上海高可信计算重点实验室(华东师范大学) 上海 200050)
[4](内蒙古师范大学计算机科学与技术学院 内蒙古 010011)
(jindm@stu.pku.edu.cn)

ChatModeler: A Human-Machine Collaborative and Iterative Requirements Elicitation and Modeling Approach via Large Language Models

Jin Dongming[1,2], Jin Zhi[1,2], Chen Xiaohong[3], and Wang Chunhui[4]
[1](*School of Computer Science*, *Peking University*, *Beijing* 100871)
[2](*Key Laboratory of High Confidence Software Technologies* (*Peking University*) , *Ministry of Education*, *Beijing* 100871)
[3](*Shanghai Key Laboratory of Trustworthy Computing* (*East China Normal University*), *Shanghai* 200050)
[4](*College of Computer Science and Technology*, *Inner Mongolia Normal University*, *Inner Mongolia* 010011)

Abstract Requirements elicitation and modeling is a critical step in requirements engineering that affects subsequent system design and implementation. Traditional requirements elicitation and modeling methods usually involve multiple types of stakeholders, such as requirement providers and requirements analysts, working together in an iterative manner, which requires a lot of manpower. It is important to reduce the burden of requirements providers and requirements analysts and improve the efficiency of elicitation and modeling. Some of the existing efforts use knowledge bases to provide more knowledge to assist modeling or elicitation, and some use natural language processing techniques to automate the elicitation or modeling process. But they do not reduce the burden of requirements providers. Leveraging the generative capability of LLMs (large language models), we provide ChatModeler, a requirement elicitation and modeling framework for iterative human-machine collaboration. Specifically, based on the division of labor and collaboration relationship of real-world requirements teams, some work of requirements providers and requirements analysts is taken up by LLMs, while requirements providers only need to perform confirmation. In this paper, prompts are designed for the various roles played by the various LLMs, which varies with the meta-model of the requirements. ChatModeler is used in 14 sets of comparative experiments with state-of-the-art automated modeling approaches for three types of requirement models on seven requirement cases, demonstrating the superiority of ChatModeler in both reducing the number of interactions of the requirements provider and generating higher quality requirement models.

Key words requirements engineering; requirements elicitation; large language models; human-machine collaboration; prompt engineering

摘 要 需求获取和建模是需求工程中的关键步骤,影响后续系统设计与实现.传统的需求获取和建模方

收稿日期: 2023-09-12; **修回日期**: 2023-12-12
基金项目: 国家自然科学基金项目(62192730, 62192731, 62162051)
This work was supported by the National Natural Science Foundation of China (62192730, 62192731, 62162051).
通信作者: 金芝(zhijin@pku.edu.cn)

法通常由需求提供者、需求分析师等多类干系人共同协作、反复迭代完成，需要耗费大量的人力. 如何减轻需求提供者与需求分析师的负担、提高获取和建模的效率有着重要意义. 现有工作中有的使用知识库来提供更多知识，以辅助获取或者建模，有的利用自然语言处理等技术对获取或者建模过程进行自动化，但是它们并没有减轻需求提供者的负担. 利用大语言模型（large language models, LLMs）的生成能力，提供了一种人机协作的迭代式需求获取和建模框架 ChatModeler.具体来说，根据真实世界中需求团队的分工及协作关系，将部分需求提供者、需求分析师等角色的工作由大语言模型承担，而需求提供者只需要进行确认. 为大语言模型扮演的各种角色进行了提示词设计，该提示词会随需求的元模型而变化. ChatModeler 在 7 个需求案例上与 3 种需求模型的自动建模方法进行了 14 组对比实验，证明了 ChatModeler 在降低需求提供者的负担和生成高质量需求模型 2 个方面上的优越性.

关键词 需求工程；需求诱导；大语言模型；人机协作；提示工程

中图法分类号 TP311

需求获取和建模是指需求分析师从需求提供者获取需求，并据此构建需求模型的过程. 传统的需求获取方法需要需求分析师和需求提供者不断地沟通，直到需求提供者确认所有的需求，这是一个迭代的过程[1]. 随着软件系统的规模和复杂度的不断增大，涉及的干系人（即角色）和需求的数量越来越多，许多项目的需求文档通常有数百页. 在这样的背景下，传统的需求获取方法由于需要干系人之间不断地沟通和持续地迭代，给需求提供者和需求分析师造成巨大的负担. 因此如何提高需求获取和建模的效率成为亟待解决的问题.

很多研究者提出了自动化需求获取与建模的方法[2-4]，主要包括 2 种：一种是基于知识库的方法，即将需求工程中的领域知识和经验封装在知识库中，利用知识库中的相关知识引导进行需求获取[5-7]. 另一种是基于机器学习的方法，即设计神经网络从自然语言需求描述中提取相应的建模元素，并人工或者设计规则组装成需求模型[8-10]. 然而，这 2 种方法仅能自动化需求获取和建模过程中的部分步骤，并不能替代需求干系人，尤其是需求提供者的大部分工作. 如何进一步减轻需求提供者和需求分析师的工作负担，提高获取与建模效率，仍是一个待解决的问题.

最近，由于以 ChatGPT 为代表的大语言模型（large language models, LLMs）在软件工程的一系列任务上取得了优异的性能[11]. 大语言模型汇聚了丰富的人类知识，不仅具有强大的自然语言生成和对话能力，而且可以通过指令学习[12]转变成领域“专家”，具有扮演需求提供者、需求分析员等多类干系人的潜力. 因此，有可能使用大语言模型（部分）替代需求提供者和需求分析师的工作，加速迭代式需求获取和建模的过程. 其中要解决的主要问题是如何通过人机交互引导大语言模型生成对需求获取和建模有帮助的信息[13].

受现实世界需求分析流程[14]和团队合作方法[15-16]的启发，我们提出了利用大语言模型进行需求获取和建模的人机协作迭代框架 ChatModeler.具体而言，首先明确需求获取和建模过程中参与交互的干系人，如收集者、建模师和检测员；然后利用大语言模型指令学习的能力设计不同角色的提示词以扮演需求团队中的不同角色，组建虚拟的需求团队. 这种方法只要求需求提供者进行确认或者少量输入，可以极大地减轻需求提供者和需求分析师的负担，达成通过多次人机交互进行高效需求获取和建模的目的.

为了验证 ChatModeler 的效果，我们使用 2 种大语言模型在 7 个公开用于需求建模的案例上与 3 种需求模型，如用例图、序列图和问题图[17]的先进自动建模方法，如 IT4RE 方法、GSQ 方法和 GPD 方法进行了对比实验. 评估结果显示 ChatModeler 能够有效降低需求提供者的交互负担，能够生成质量更高的需求模型. 最后，我们用一个真实的案例，即太阳搜索控制系统进行了案例研究，证明了其可行性.

本文的主要贡献包括 3 个方面：

1）提出了一个用于需求获取和建模的人机协作迭代框架，该框架使用多个大语言模型进行角色扮演，能替代需求提供者和需求分析师的大部分工作，提高了需求获取和建模的效率.

2）设计了一种基于大语言模型和需求元模型的协作机制，通过提示词定义了团队角色及其之间的协作过程，支持需求的获取和多种需求模型的构建.

3）进行了方法评估和案例研究，方法评估证明了 ChatModeler 能降低需求提供者的负担，并能生成

效果更好的需求模型,案例研究证明了其可行性.

本文其余部分的组织如下.第一节对相关工作进行了比较.第二节介绍了大语言模型和需求模型等预备知识.第三节说明了我们提出的框架和设计细节.第四节报告了我们的研究问题和实验结果.第五节展示了我们的案例研究结果.第六节讨论了我们工作的意义和具有的局限性.第七节对本文进行了总结,并说明了未来的工作内容.

1 相关工作

相关工作主要体现在迭代式需求获取和需求建模中部分步骤的自动化,主要包括基于机器学习的自动化以及基于知识库和大语言模型的自动化.

1.1 基于机器学习的自动化建模

根据设计方法的不同,该研究工作又可以分为两大类别:基于自然语言处理的统计方法和基于神经网络的深度学习方法.

1)基于自然语言处理的统计方法主要是借助已有的自然语言处理工具对软件需求文档进行语义或者语法方面的计算,进而设计规则对建模元素进行提取.例如,Elbendak 等人[8]基于文本的预处理技术设计了需求提取工具 Class-Gen,该工具能够从自然语言需求文档中提取类和对象,进而生成用例图模型.Jahan 等人[3]基于词性和依存关系提出了一种从英语用例描述中自动生成序列图的方法.文献 [3, 11] 所提方法的设计过程繁琐且领域迁移能力弱.

2)基于神经网络的深度学习方法主要是使用深度学习技术设计网络模型,以从需求文本中提取相应的建模元素.例如,Pudlitz 等人[9]使用基于双向长短时记忆网络和卷积神经网络的命名实体识别模型从软件需求文本中提取目标对象的状态信息.Qian 等人[10]设计了一种层次神经网络方法,从需求描述中提取过程模型中的动作、控制流等建模元素.文献 [9−10] 所提方法可以提高领域迁移能力.

尽管文献 [3, 11, 12−13] 所提方法都达成了较好的效果,能够辅助建模者进行需求建模,但是它们都是以需求文档或者需求描述为输入,一次性产生各种建模元素,且在过程中无法修改和迭代系统需求.如果自动化提取过程中遗漏了部分建模元素,建模者只能人工检查需求文档进行修改,迭代修改的过程也需要花费大量时间人工手动来完成.

1.2 基于知识库和大语言模型的自动化获取和建模

基于知识库的方法是通过用软件开发者积累的经验或者领域分析的结果,辅助需求的获取和建模.例如,Dardenne 等人[5]提出了 KAOS 项目的元模型制导下的需求获取方法.金芝[6]以企业信息系统为研究背景,提出了一种基于本体的需求获取方法,有利于保证需求获取的完整性.文献 [7, 18−19] 以问题框架方法中的元模型——问题框架本体,引导需求提供者进行问题图的建模以及需求规约的获取.

随着大语言模型能力的不断增强,引发了需求工程领域社区越来越多的兴趣.White 等人[2]探究了将大语言模型用于需求获取、原型设计、自动设计等各项任务的提示词,但是该模型处理各项任务时相互独立,没有考虑到不同任务在不同阶段会相互反馈、相互影响.Xing 等人[17]利用大语言模型在软件工程的各项任务中创建了思维链,但是对需求任务考虑较少.

在文献 [5−7, 18−20] 方法中,基于知识库的方法仅仅是检索知识,并不具备生成知识库外新需求的能力.这类方法依赖于检索知识库中积累的有限的经验,不支持迭代式获取.此外,虽然研究者已经开始探索大语言模型在需求工程方面的应用,但目前仅限于设计单个任务的提示词,没有考虑利用其生成能力进行不同角色间交互式协作迭代.

2 预备知识

在本节中,我们主要介绍大语言模型、提示工程以及需求元模型 3 个方面的预备知识.

2.1 大语言模型和提示工程

大语言模型的兴起可以追溯到 Transformer[21] 模型以及 BERT[22] 模型的发布.之后,随着 GPT 系列模型的出现(如 GPT-3[23]),大语言模型在一系列任务上取得了优异的性能[24-26].最近,以 ChatGPT 为代表的大语言模型能够通过指令或者提示词[27]来完成用户的各种文本处理任务[28].

大语言模型聚集了丰富的人类知识,具有强大的自然语言生成能力和对话能力,它可以生成与用户进行自然对话的响应,因此大语言模型可以模拟人类对话过程,与人类进行交互.大语言模型还具有上下文感知能力,能够根据当前的回答记录引导用户提供更加详细的需求信息.

使用大语言模型的关键是设计提示词.提示词的质量是大语言模型在处理下游任务上性能好坏的重要因素[25].因此,需要为大语言模型设计高质量的提示词以获得预期的模型输出.

2.2 需求元模型

需求建模方法有很多种，如结构化分析方法[29]、面向对象方法[30]、问题框架方法[17]、面向目标的方法[5]等. 每种建模方法都有其需求元模型，即建模概念及其之间的关联关系. 下面以问题框架方法中的问题图为例来说明需求元模型. 图 1 展示了问题框架方法的需求元模型.

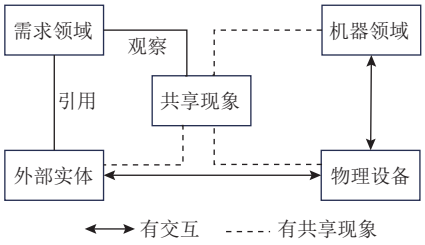


Fig. 1 Requirement meta-model of problem frame

图 1 问题框架方法的需求元模型

如图 1 所示，问题框架方法中的概念包括机器领域(MD)、需求领域(RD)、外部实体(OE)、物理设备(PD)和共享现象(SP). 表 1 给出了这些概念的定义及举例说明. 其中，机器领域为待开发的软件；外部实体是不能直接与机器领域交互的实体；物理设备则是可以与机器领域直接交互的实体，机器领域通过物理设备与外部实体进行交互；机器领域与物理设备之间存在共享现象，而物理设备与外部实体之间存在共享现象. 需求领域则是需求的希求式表达，它引用或者约束外部实体.

Table 1 Entity Modeling Elements in Problem Frame Method

表 1 问题框架方法中的实体建模元素

实体元素	含义	举例说明
机器领域	指待开发软件	太阳搜索控制系统
需求领域	希望看到的效果	追踪太阳
外部实体	现实中的环境实体	太阳、卫星
物理设备	现实中的物理设备	太阳敏感器
共享现象	领域间共享的数据	太阳可见标志

在需求元模型的基础上，问题框架方法定义了需求模型——问题图，它是需求元模型的实例. 图 2 展示了太阳搜索控制系统的一个问题图. 该系统主要是通过采集各种硬件设备的数据以确定卫星当前的姿态，进而实现追踪太阳.

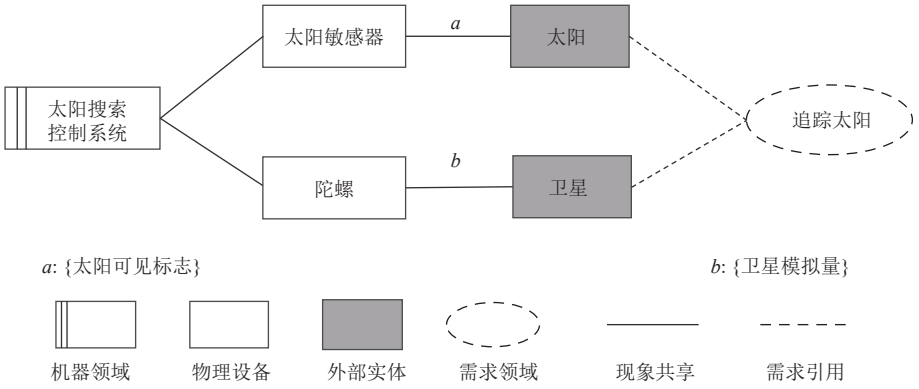


Fig. 2 Part problem diagram of the sun search control system

图 2 太阳搜索控制系统的部分问题图

从图 2 可以看出，问题图中的节点表示需求元模型中的实体概念，包括机器领域、物理设备、外部实体和需求领域等. 边表示了实体概念之间的关联关系. 实体概念间的关系可以分为两类，一类是需求引用关系，另一类是现象共享关系. 需求引用关系被建模成二元关系，多存在于需求领域与外部实体之间，表示期望在问题领域上出现的效果. 本文用二元组(e_1, e_2)表达这种关系，其含义是指为了实现 e_2 的需求效果，系统需要涉及 e_1 这个外部实体. 现象共享关系被建模为三元关系，多存在于机器领域与物理设备或者物理设备与外部实体之间，表示 2 个领域实体之间接收或者发送某些数据或者指令. 本文使用三元组$(e_1, e_2, data)$来表示这种关系，其含义是指 e_1 实体和 e_2 实体之间存在数据 *data* 的传递.

3 人机协作的需求获取和建模框架

本节给出基于大语言模型的人机协作迭代式需求获取和建模框架. 首先给出该框架的角色组成和迭代关系，然后介绍了协作框架中不同角色之间的协作流程.

3.1 ChatModeler 整体框架

现实世界中，需求团队进行需求获取和建模的一般流程为：首先需求收集者与需求提供者进行对

话以交谈待开发系统的需求;然后将对话记录整理成需求片段;接下来需求建模者阅读整理后的需求片段,从中提取建模元素进行需求建模;最后需求检测员阅读建立的需求模型以检测需求的完备性. 如果发现缺少某项需求建模元素,则给需求收集者反馈缺失信息. 需求收集者需要根据反馈的信息再次与需求提供者进行访谈,补充更多的需求信息.

受上述现实世界中需求团队工作流的启发,本文提出了基于大语言模型的人机协作迭代式需求获取和建模框架 ChatModeler, 如图 3 所示. 其中需求提供者依然是现实世界中的需求提供者,但是他/她的重要职责由原来的提供需求转换为确认需求,而提供需求的职责大部分转交给了大语言模型扮演的需求收集者. 同时,原来需求工程师(包括需求收集者、建模师、检测员)的工作则由 3 个大语言模型组成的需求分析团队负责.

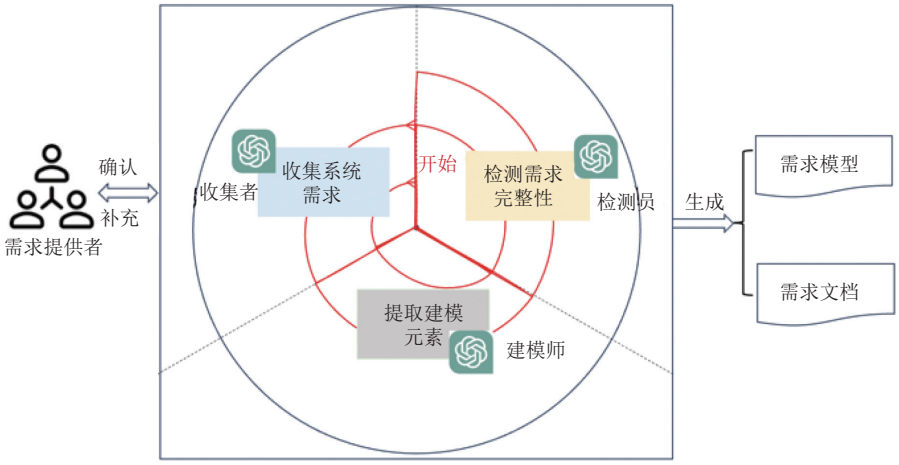


Fig. 3 Human-machine collaborative framework for requirement elicitation and modeling

图 3 人机协作的需求获取与建模框架

在该协作框架中,3 个大语言模型分别扮演不同的角色,即收集者、建模师和检测员. 它们之间相互协作,与需求提供者一起迭代式进行需求获取和建模. 在该框架中,我们设计了这 4 种角色之间的协作流程,如图 4 所示.

在该协作流程中,首先是需求提供者和收集者进行 n 次对话以收集需求提供者对待开发系统的需求;其次收集者将本轮的 n 次对话内容整理成一个需求片段,并将该需求片段发送给建模师;然后建模师从该需求片段中提取目标需求模型的建模元素,并将提取的建模元素发送给检测员;随后检测员根据当前提取的建模元素检测需求的完整性,并将本轮检测的缺失信息发送给收集者;最后收集者根据收到的缺失信息,进入下一轮需求迭代过程,再次与需求提供者进行对话.

迭代多轮后,收集者收到检测员发送的需求检测通过的信息时,收集者将整理与需求提供者的对话记录,按照设定的需求文档模板自动生成需求文档. 建模师会输出在多轮迭代过程中提取的全部目标需求建模元素.

3.2 基于大语言模型的协作团队定义

这一步的目标是根据提出的需求获取和建模框架定义协作团队,为大语言模型设计提示词以实现大语言模型承担不同角色的分工和具备协作关系. 主要任务包括团队任务定义与角色定义 2 方面.

3.2.1 团队任务定义

根据 3.1 节设计的角色之间的协作流程,我们定义了用于协作式需求获取和建模任务的团队提示词模版. 具体来说,设计的团队提示词(Prompt)模版由 4 部分组成,分别是团队说明(Team)、职能说明(Role)、行为说明(Action)和交互信息(Message). 其中,团队说明是用来阐述团队的构成以及提示不同角色之间需要相互协作;职能说明是提示大语言模型需要扮演的角色以及这个角色的职责和分工;行为说明是提示大语言模型具备的交互行为,即要扮演的这个角色会收到来自哪些角色的信息以及收到这个信息后需要执行怎样的行为;交互消息是提示大语言模型当前来自其他角色的具体交互信息. 这 4 部分提示词的设计具体如表 2 所示.

3.2.2 角色定义

从 3.1 节设计的角色之间的协作过程中,我们首先得到大语言模型扮演的 3 种角色的职责和行为,再进一步进行提示词设计,由此定义角色.

1)收集者角色的定义

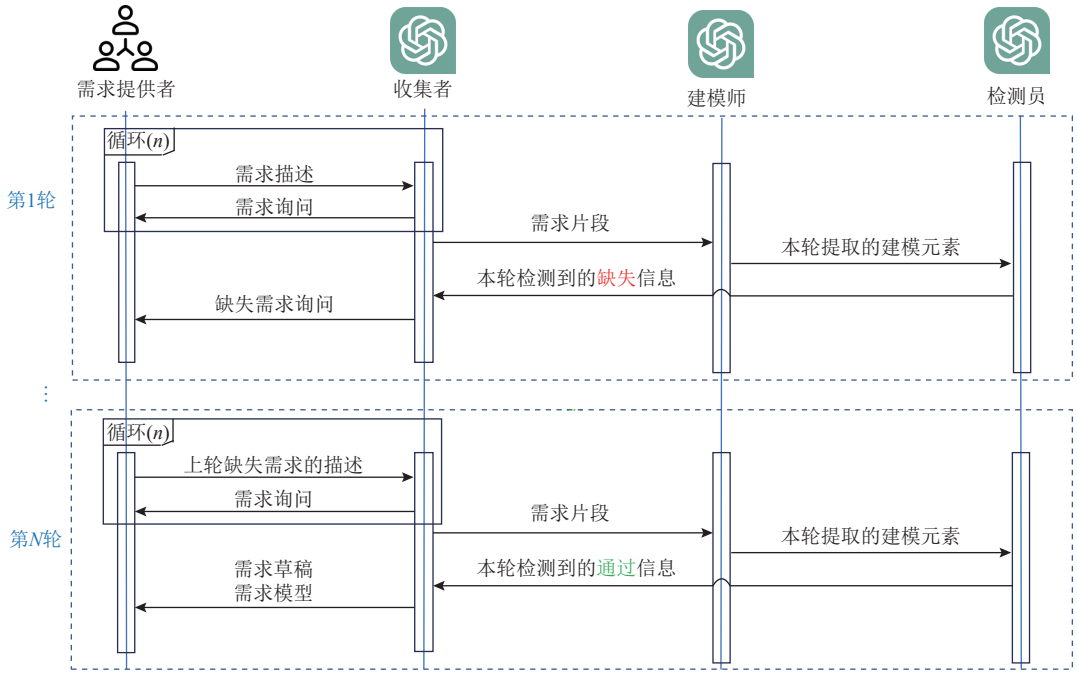


Fig. 4 Sequence diagram of collaboration relationships among four roles

图 4 4 种角色之间的协作关系序列图

Table 2 Prompt Template for Different Roles in the Requirements Elicitation and Modeling Team

表 2 需求获取和建模团队中不同角色的提示词模板

参数	具体说明
Team	现有一个需求获取和建模团队，包括 1 个需求收集者、1 个需求建模师、1 个需求检测员，不同人员之间分工不同，相互合作.
Role	我希望你能在这个需求团队中扮演一个<Role>，负责<Duty>. 你会收到来自<Team>的<Message>.
Action	如果你收到<Team>的<Message>，请做<Action>.
Message	来自<Team>的<Message>的内容是<Content>.

注: Prompt = Team + Role + Action + Message. Role 表示大模型扮演的角色，Duty 表示角色的职责，Team 表示发生交互的其他角色，Message 表示交互角色会发送什么信息，Action 表示角色执行的动作，Content 表示来自交互角色的反馈内容. 这些都是提示词模板的参数.

Table 3 Parameter Values of Collector Prompt

表 3 收集者提示词的参数取值

参数	取值
Role	需求收集者
Duty	与需求提供者沟通需求
Team	需求检测员
Message	待补充的缺失需求
Action	向需求提供者提问
Content	当前缺少<MissingReq>，请询问

根据我们设计的协作框架，收集者使用大语言模型的对话能力引导需求提供者，输入更加详细的需求描述，负责与需求提供者进行多轮对话，收集需求提供者对系统的期望，并据此生成需求片段. 收集者也会收到来自检测员反馈的待补充的需求信息，并根据反馈信息向需求提供者执行下一次对话以沟通需求，循环往复，逐步完善需求. 因此，根据上述收集者的角色和行为定义，可以将表 2 提示词模版中的参数进行实例化以获得收集者的提示词. 收集者实例化的提示词模版参数具体如表 3 所示. 其中 MissingReq 表示需求检测员反馈的缺失需求信息.

此外，当需求收集完成后，收集者需要梳理当前所有的会话记录，并将其整理成一个完整规范的需求文档. 但是大语言模型输出的需求文档的格式具有一定的随机性，每次输出的需求文档遵循的规范不同，不利于人工阅读. 因此，本文设计了一个需求文档的统一规范格式，然后以提示词的形式将需求文档模版知识输入给大语言模型. 具体的提示词如表 4 所示.

该需求模版包括概述、系统组成和功能需求 3 部分. 概述部分介绍系统的基本情况，包括目标等；系统组成部分介绍系统包含的物理设备；功能需求部分介绍为了达成每个功能需求的动作序列. 表 4 的提示词中包含了每部分推荐的句型.

2）建模师定义

根据我们设计的协作过程，建模师的职责是理解需求提供者整理的需求片段，并从中提取需求元

Table 4 The Prompt for Requirements Documents Generation

表 4 需求文档生成的提示词

请你以下面的模版格式输出相应的需求文档，注意不要输出其他内容.
#1.概述
<系统名称>需要能够<系统目标>.
#2.系统组成
<系统名称>由<物理设备集合>组成.
#3.功能需求
* 为了实现<需求领域>：
<机器领域>使用<物理设备><感知/采集/接收/控制><外部实体>的<共享现象>.

注: Prompt = Explain + Document.

Table 5 The Prompt for the Modeler

表 5 建模师的提示词

我希望你能在这个需求团队中扮演一个需求建模师，负责从需求收集者提供的需求中提取建模元素. 你会收到来自需求收集者的需求片段. 具体要提取的建模元素及含义为： 机器领域：待开发软件. 外部实体：外部世界的实体. 物理设备：一些外部世界的设备. 需求领域：预期达到的效果. 共享现象：领域间共享的数据. 需求引用关系 ：存在于需求领域和外部实体之间，表示期望在机器领域上出现的效果. 现象共享关系： 存在于机器领域、物理设备和外部实体之间，表示接收或发送数据或指令.
如果你收到需求收集者的需求片段，请立即提取建模元素.
来自需求收集者的需求片段的内容是<Content>

注: Content 表示当前迭代阶段来自需求收集者的需求片段.

模型中的需求建模元素. 建模师会将提取出来的建模元素发给检测员. 考虑到需求建模元素中存在各种专有名词，大语言模型不能准确理解其含义. 因此，本文在提示词中添加了需求元模型中对应的各种建模元素的含义说明，辅助大语言模型理解这些专有名词和需求建模元素的概念.

由于需求建模与具体的需求模型密切相关，提示词的设计就需要考虑到目标建模的需求模型，即建模师的提示词是由目标建模的需求元模型制导产生的. 下面以 2.2 节列出的问题框架方法[28] 中问题图的需求元模型作为建模师设计提示词来说明如何基于需求元模型进行建模师提示词的设计. 构建的建模师的提示词如表 5 所示.

在表 5 中，建模师的提示词包含了图 1 问题图需求元模型中的各种建模元素以及含义说明. 如果对其他需求模型进行建模，如面向结构的方法、面向对象的方法等，只需要根据目标建模的需求元模型修改该提示词中建模元素的含义说明即可.

3)检测员定义

检测员的职责主要是基于建模师提取出的建模元素和需求元模型检测缺失的需求信息，反馈缺失信息以指导下一轮需求收集者的需求获取过程. 需求元模型对需求完整性的检查具有一定的指导作用. 我们以图 1 的需求元模型为例，为扮演检测员的大语言模型设计了 10 条需求完整性的检查规则.

我们将这 10 条规则按照检测员检查的顺序进行了编号，并按照编号顺序写到提示词中，提示大语言模型按照编号从上往下检查需求. 具体的提示词如表 6 所示. 这 10 条检查规则与图 1 密切相关. 前 7 条规则是按照需求元模型中需要包含的建模实体元素如机器领域、需求领域、外部实体、物理设备和共享现象的顺序排列. 因为问题框架需求元模型的需求获取逻辑依次是要开发怎样的系统、开发这个系统的意图是什么、为了实现这样的意图需要监测哪些外部实体、这些外部实体是通过哪些物理设备监测的、物理设备需要采集这些外部实体的什么数据或者指令. 后 3 条规则是需求元模型中需要包含的建模关系元素，并按照先需求引用、后现象共享排序.

Table 6 The Prompt for the Checker

表 6 检测员的提示词

我希望你能在这个需求团队中扮演一个需求检测员，负责从需求建模师提取的建模元素中检查需求的完整性. 并返回缺失的需求. 你会收到来自需求建模师提取的建模元素. 请按照以下规则从上往下依次检查需求的完整性. 如果全部满足，则反馈需求完整. 规则 1：机器领域有且仅有一个. 规则 2：需求领域至少要有一个. 规则 3：每个需求领域至少要引用或者约束一个外部实体或者物理设备. 规则 4：每个外部实体都通过物理设备与机器领域相连. 规则 5：每个物理设备都要与机器领域相连. 规则 6：每个外部实体与相连的物理设备之间至少存在一个共享现象. 规则 7：机器领域与相连的物理设备之间至少存在一个共享现象. 规则 8：每个需求领域实体至少与一个物理设备或外部实体之间存在需求引用关系. 规则 9：每个外部实体至少要与一个物理设备之间存在共享现象关系. 规则 10：机器领域要与物理设备之间存在共享现象关系.
如果你收到需求建模师提取的需求建模元素，请立即根据上述规则检查需求的完整性.
来自需求建模师提取的需求建模元素的内容是<Content>

4 实验设计与结果分析

本文利用大语言模型的语义理解和生成能力提出了需求获取和建模协作框架 ChatModeler. 与其他方法相比，该方法能与需求提供者发生较少的交互，并生成质量较高的需求模型. 为了证明这个效果，设计了 2 个研究问题：

问题 1：ChatModeler 在获取需求时能否减轻需求提供者的负担，降低与需求提供者的交互. 问题 1 旨在证明 ChatModeler 在获取需求方面降低了与需求提供者的沟通负担. 我们在 7 个案例上计算了 2 种大语言模型下的 ChatModeler 与需求提供者发生交互的次数，并与 3 种需求模型的自动化方法进行了对比实验.

问题 2：与其他自动建模方法相比，ChatModeler 生成的需求模型的质量怎么样. 问题 2 旨在验证 ChatModeler 生成需求模型上的效果. 我们在 7 个案例上计算了 2 种大语言模型下的 ChatModeler 生成需求模型的准确率(P)、召回率(R)和 $F1$ 得分，并与 3 种需求模型的自动化方法进行了对比实验.

4.1 实验配置

1)评估案例. 我们在 8 个公开用于需求建模的案例上评估 ChatModeler，这些案例分别是：ATM 系统需求案例(ATM)[31]；图书馆管理系统案例(TLS)[32]；自助点餐系统案例(COS)[33]；装配系统案例(TSS)[34]；时间监控系统(TMS)[35]；售货员系统(SSS)[3]；智能家居控制系统(HCS)[36]；锁链控制系统(CCS)[37].

2)基线方法. 我们选择了 3 种需求模型的需求自动化建模方法与 ChatModeler 进行对比实验. 下面对这 3 种方法进行说明.

①IT4RE 方法[38] 从需求描述中提取“参与者”和“动作”2 种建模元素，自动建模用例图.

②GSQ 方法[3] 从需求描述中提取“发起者”“接收者”和“消息”3 种建模元素，自动建模序列图.

③GPD 方法[18] 是从需求描述中提取“问题领域”“需求领域”等多种建模元素，自动建模问题图.

3)大语言模型的选取. 本文实验使用了 2 个大语言模型，一个是闭源模型 GPT-3.5，另一个是开源模型 ChatGLM-6B.GPT-3.5 模型是 OpenAI 公司开发的，该模型能够遵循提示指令提供详细的响应信息. 本文实验是通过 LangChain 框架[39] 提供的 API 访问模型. 考虑到模型的回答具有一定的随机性，为了保证实验结果的稳定性，我们使用 gpt-3.5-turbo 作为我们的基本模型，同时也将 gpt-3.5-turbo 模型的解码温度设置为 0. ChatGLM-6B 模型是清华大学开源的、支持中英双语的对话模型，能够生成符合人类偏好的回答[40]. 本文实验是将开源的模型通过 API 的方式进行本地部署，然后利用 LangChain 提供的自定义大语言模型的方式将模型进行本地部署和 API 封装.

4)评价指标. 为了验证 ChatModeler 协作框架能够降低需求提供者的负担，提高需求获取和建模的效率，本文选择在需求获取过程中与需求提供者发生的交互次数作为衡量指标. 为了衡量 ChatModeler 生成的需求模型的质量，本文选择生成需求模型元素的准确率、召回率和 $F1$ 得分作为衡量指标. 准确率是指正确生成的需求模型元素数量与生成的全部需求模型元素数量之比. 召回率是指正确生成的需求模型元素数量与应该被生成的需求模型元素数量之比. $F1$ 是平衡准确率和召回率的综合评价指标.

4.2 问题 1 的设计和结果

1)实验设计

为了评估 ChatModeler 能否降低与需求提供者发生交互的次数，本文选择将其与 3 种先进的需求自动建模方法 IT4RE、GSQ 和 GPD 在 7 个案例上进行对比实验，分别计算与需求提供者的交互次数. 对比的 3 种自动化需求建模方法的选取涉及了多种需求建模原型，以验证 ChatModeler 在各种需求模型中均适用.

2)实验过程

对于传统方法与需求提供者交互次数的计算，考虑到这些方法对需求的输入是一次性的且输入的是整理好的需求文档，无法直接计算得到此需求时与提供者发生交互的次数. 因此，本文选择将传统方法处理的需求描述的句子数目视为交互的次数. 因为我们认为这些案例中的需求描述是简洁完备的. 对于 ChatModeler 与需求提供者交互次数的计算，是在人工理解需求描述后，与 ChatModeler 进行人机对话，将需求提供者输入需求的次数视为产生交互的次数.

首先，使用 IT4RE 和 ChatModeler 分别处理 4.2 节中的前 5 个案例以获取和建模用例图，并分别计算和对比这 2 种方法的交互次数. 其次，使用 GSQ 和 ChatModeler 处理 SSS 案例以获取和建模序列图. 最后，使用 GPD 和 ChatModeler 处理 HCS 案例以获取和建模问题图.

3)实验结果及分析

各种方法在获取需求时与需求提供者产生交互的次数如表 7 所示. 从表 7 可以看出，ChatModeler 能够进一步降低与需求提供者交互的次数. ChatModeler 在处理 7 个案例上均低于现有 3 种先进需求建模方法. 与 IT4RE 相比，ChatModeler 在处理前 5 个案例时，与需求提供者交互的次数平均降低了 34%. 与 GSQ 相比，ChatModeler 在 SSS 案例上，与需求提供者交互的次数降低了 21.4%. 与 GPD 相比，ChatModeler 在 HCS 案例上，与需求提供者交互的次

Table 7 Numbers Interacted with Requirement Providers

表 7 与需求提供者发生交互的次数

案例情况		其他方法		ChatModeler	
				GPT-3.5	ChatGLM-6B
需求模型	案例	方法名称	交互次数	交互次数	交互次数
用例图	ATM	IT4RE	9	5	5
	COS		8	7	6
	TLS		15	10	9
	TAS		7	4	6
	TMS		11	5	8
序列图	SSS	GSQ	14	12	10
问题图	HCS	GPD	10	8	8

数降低了 20%.

ChatModeler 适用于多种需求模型的需求获取. ChatModeler 与用例图、序列图和问题图的先进需求建模方法相比, 与需求提供者交互的次数均有所下降. 对于建模用例图、序列图和问题图, 使用 GPT-3.5 大语言模型的 ChatModeler 与需求提供者交互的次数分别降低了 38%、14.3% 和 20%.

ChatModeler 在多个大语言模型上都能降低与需求提供者交互的次数. 在 7 个案例上, GPT-3.5 和 ChatGLM-6B 这 2 种大语言模型实现的 ChatModeler, 与需求提供者产生交互的次数分别平均降低了 31.1% 和 29.7%. 由此可以看出, 这 2 种大语言模型下交互次数均有减少, 而且 GPT-3.5 大语言模型实现的 ChatModeler 的效果略好. 因此, ChatModeler 是一种协作框架, 适用于多种大语言模型, 但是我们建议使用最新的大语言模型.

4.3 问题 2 的设计和结果

1)实验设计

为了评估 ChatModeler 生成的需求模型的质量, 本文同样选择将其与 4.2 节中的 3 种先进需求自动建模方法 IT4RE、GSQ 和 GPD 对 7 个案例进行对比实验, 分别计算从需求描述中产生需求建模元素的准确率、召回率和 $F1$ 值.

2)实验过程

对于 4.2 节中的每一个案例, 首先由人工理解需求描述; 其次与 ChatModeler 进行对话交互, 生成人工与 ChatModeler 的对话记录; 然后对建模师每个迭代周期产生的需求建模元素合并; 最后计算 ChatModeler 产生的需求建模元素与的准确率、召回率和 $F1$ 值.

3)实验结果及分析

ChatModeler 与 3 种传统方法生成的需求模型的质量评估结果如表 8 所示. 从表 8 可以看出, ChatModeler 协作框架在多个案例上的效果均高于传统方法. 与 IT4RE 相比, ChatModeler 在前 5 个案例上的召回率均保持或提升, 准确率在 TAS 案例上略有下降, 但在其他 4 个案例上分别相对提升了 18.1 个百分点、12.6 个百分点、8.3 个百分点、22.8 个百分点. 与 GSQ 相比, ChatModeler 在 SSS 案例上的准确率有所下降, 但是召回率相对提升了 18.9 个百分点, 综合评价指标 $F1$ 值相对提升了 5.1 个百分点. 与 GPD 相比, ChatModeler 在 HCS 案例上的准确率、召回率和 $F1$ 值分别相对提升了 9.6 个百分点、49.3 个百分点和 26 个百分点.

ChatModeler 协作框架相对于多种需求模型生成质量均有所提升. 对于用例图建模, ChatModeler 处理前 5 个案例上的平均准确率、召回率和 $F1$ 值分别相对提升 13.2 个百分点、8.5 个百分点和 11.9 个百分点. 对于序列图和问题图建模, ChatModeler 在处理 SSS 案例和 HCS 案例时, 召回率和 $F1$ 得分均有所增加.

ChatModeler 协作框架在多个大语言模型上都能提高生成需求模型的质量. 对于 GPT-3.5 大语言模型,

Table 8 Evaluation Results of ChatModeler on Modeling Quality

表 8 ChatModeler 在建模质量上的评估结果

案例情况		其他方法				ChatModeler					
						GPT-3.5			ChatGLM-6B		
需求模型	案例	方法名称	准确率/%	召回率/%	$F1$ 值/%	准确率/%	召回率/%	$F1$ 值/%	准确率/%	召回率/%	$F1$ 值/%
用例图	ATM	IT4RE	50	83	63	62.5	83.3	71.4	55.6	83.3	66.7
	COS		63	56	59	69.2	90	78	72.7	72.7	72.7
	TLS		50	56	53	58.3	77.8	66.7	50	75	60
	TAS		17	100	29	14	100	25	14.3	100	25
	TMS		56	100	71	100	100	100	37.5	75	50
序列图	SSS	GSQ	87.1	75	80.6	81.6	86.1	83.8	80	92.3	85.7
问题图	HCS	GPD	76.4	58.3	65.7	85.7	92.3	83.8	81.8	81.8	81.8

ChatModeler 在 7 个案例上的平均准确率、召回率和 $F1$ 值分别相对提升 17.9 个百分点、19.2 个百分点和 20.8 个百分点. 对于 ChatGLM-6B 大语言模型, ChatModeler 在 7 个案例上的平均准确率相对降低了 1.9 个百分点, 但是召回率和 $F1$ 值分别相对提升 9.8 个百分点和 1.7 个百分点.

5 案例研究

在本节, 我们使用一个实际的开发系统进行了案例分析和研究, 以证明本文设计方法的实用性. 我们选择的案例是来自航空航天控制领域的一个真实开发系统, 即太阳搜索控制系统[41-42]. 该系统主要是通过采集各种硬件设备的数据以确定卫星当前的姿态, 进而实现捕获太阳. 我们选定该案例, 一方面是因为该案例真实面临着需求获取和建模的任务, 另一方面是因为它属于嵌入式系统, 需求相对复杂, 而且需要与现实世界中的实体紧密联系, 适合采用问题框架方法建模.

5.1 案例研究过程

本文案例研究过程中使用的大语言模型是 gpt-3.5-turbo, 模型的设置和本文实验环节相同, 具体设置如 4.1 节所述. 在案例研究过程中, 我们首先阅读了太阳搜索控制系统的系统需求文档, 然后人工与 ChatModeler 进行多轮迭代式对话交互. 完整的人机对话记录以及由大语言模型扮演的 3 种角色生成的需求片段、需求模型和需求草稿可以在链接中访问①.

对话结束后, 将建模师生成的建模元素人工组装成问题图, 并将表 4 的提示词输入给收集者以根据对话记录生成相应的需求文档.

5.2 案例结果

1)生成的问题图

图 5 展示了从对话记录中生成的问题图. 我们让专业的需求建模工程师人工阅读了人机对话记录, 然后评估建模师生成的问题图. 经过人工评估发现, ChatModeler 能够正确提取太阳、卫星等外部实体, 也能够正确识别出追踪太阳、接收操作员的指令 2 个需求领域, 同时也能识别出监测外部实体的物理设备以及它们之间互相的数据传送信息, 如太阳敏感器需要感知太阳的可见标志, 推力器需要采集卫星的角度等.

2)生成的需求文档

表 9 展示了 ChatModeler 生成的太阳搜索控制系统的需求文档.

对生成的需求文档人工评估发现, 概述部分描述出了太阳搜索控制系统的目的, 但是对意图描述不够充分; 系统组成部分准确描述了系统包含的 4 种设备; 功能需求部分准确描述了实现 2 个功能的操作序列. 所以, 我们认为大语言模型产生的需求必须由需求提供者进一步确认和修改.

6 讨 论

我们提出了利用大语言模型进行角色扮演, 人

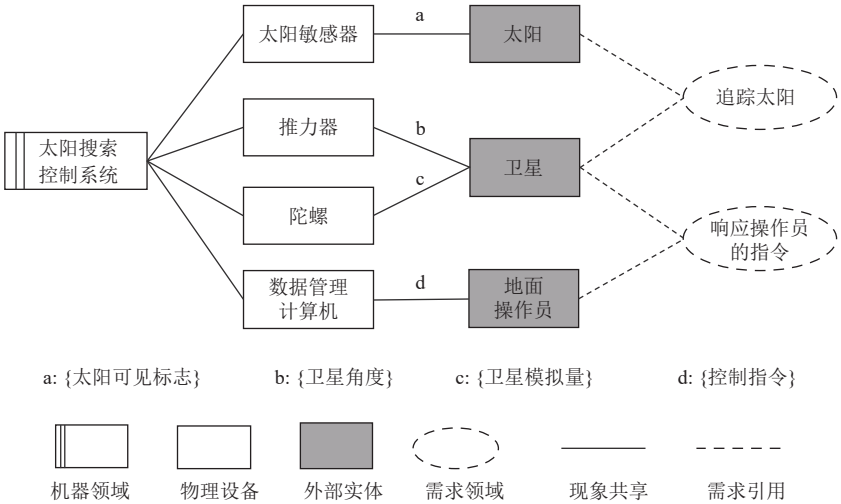


Fig. 5 Final generated problem diagram of the sun search control system

图 5 最终生成的太阳搜索控制系统的问题图

① https://github.com/publicsubmission/chatmodeler-result.

Table 9 Final Generated Requirement Document for the Sun Search Control System

表 9 最终生成的太阳搜索控制系统需求文档

1. 概述

太阳搜索控制系统需要能够感知太阳位置，控制卫星姿态以及接受地面操作员的指令，以达成准确定位和操作卫星的目的.

2. 系统组成

太阳搜索控制系统由太阳敏感器、陀螺、三轴推力器和数据管理计算机组成.

3. 功能需求

1）为了实现追踪太阳：

太阳搜索控制系统使用太阳敏感器来感知太阳的可见标志.

太阳搜索控制系统使用推力器来采集卫星的角度.

2）为了实现响应操作员的指令：

太阳搜索控制系统使用推力器来采集卫星的角度.

太阳搜索控制系统使用陀螺来采集卫星的模拟量.

太阳搜索控制系统使用数据管理计算机来接收地面操作员的指令.

机相互协作迭代式需求获取和建模框架 ChatModeler. 这种框架不仅能够降低需求提供者的负担，提高需求获取和建模的效率，而且能够生成质量更高的需求建模，提升自动化需求建模的效果. 同时，ChatModeler 框架具备良好的可扩展性，可以适用于多种需求模型的建模任务，不再局限于一种方法只能建模一种需求模型. 我们认为，ChatModeler 提供的角色扮演协同方案，也为软件工程中的其他任务提供了一种可以探索的思路.

但是，由于大语言模型本身存在幻觉问题，大语言模型可能会生成含有错误的需求文档和需求模型，且大语言模型自动化生成的需求文档和模型的质量难以保障. 本文提出的 ChatModeler 是一种半自动化的人机交互方法，大模型生成的需求草稿和模型的每个迭代阶段会由人工验证生成的内容. 如果存在错误的需求内容，会输入给大语言模型人工反馈信息.

另一方面，由于大模型上一种黑盒模型，我们目前设计的用于构建需求分析团队的提示词不可避免地会受到一些限制. 我们的协作方式也受到传统需求工程方法论的限制，没有考虑大模型时代下需求获取和建模任务下的重新分工，或许存在更加适合的虚拟需求分析团队.

7 结 论

本文提出了一种基于大语言模型的人机协作迭代式需求获取和建模框架 ChatModeler，旨在降低需求提供者的负担和生成质量更高的需求模型. 具体而言，受现实世界中需求分析团队工作流的启发，本文分析了需求分析团队中的协作流程和角色职责，进而设计提示词实现了由大语言模型扮演需求获取和建模团队. 在该团队中，大语言模型扮演的 3 种角色相互协作、不断迭代和细化需求，最终生成需求文档和需求模型. 我们进行了广泛的对比实验证明了 ChatModeler 的优越性，并在航空航天领域系统中的太阳搜索控制系统上进行了案例研究，证明了 ChatModeler 的可行性.

我们认为，多个大语言各司其职，相互协作的框架对处理需求工程中复杂分析任务具备很大的发展潜力，我们将继续在这个方向进行探索. 未来的工作主要是 2 个方面：一方面是探索更多大模型之间的分工和协作机制来处理需求工程中的获取和建模等任务；另一方面是探索如何进一步保障大模型的模型生成质量.

作者贡献声明：靳东明提出了论文思路并完成了实验；金芝对研究问题和实验方案的设计以及论文的撰写提供了帮助；陈小红和王春晖修改论文，并提出方案改进意见.

参 考 文 献

[1] Goguen J A, Linde C. Techniques for requirements elicitation [C]//Proc of the IEEE Int Symp on Requirements Engineering. Piscataway, NJ: IEEE, 1993: 152-164

[2] White J, Hays S, Fu Quchen, et al. Chatgpt prompt patterns for improving code quality, refactoring, requirements elicitation, and software design[J]. arXiv preprint, arXiv: 2303. 07839, 2023

[3] Jahan M, Abad Z S H, Far B. Generating sequence diagram from natural language requirements[C]//Proc of the 29th IEEE Int Requirements Engineering Conf Workshops (REW). Piscataway, NJ: IEEE, 2021: 39-48

[4] Wang Chunhui, Hou Lu, Chen Xiaohong. Extracting requirements models from natural-language document for embedded systems[C]//Proc of the 30th IEEE Int Requirements Engineering Conf Workshops (REW). Piscataway, NJ: IEEE, 2022: 18-21

[5] Dardenne A, Van Lamsweerde A, Fickas S. Goal-directed requirements acquisition[J]. Science of Computer Programming, 1993, 20(1-2): 3−50

[6] Jin Zhi. Ontology-based requirements elicitation[J]. Chinese Journal of Computers, 2000, 23(5): 486−492(in Chinese)
(金芝. 基于本体的需求自动获取 [J]. 计算机学报, 2000, 23(5): 486−492)

[7] Chen Xiaohong, Yin Bin, Jin Zhi. Ontology-guided requirements modeling based on problem frames approach[J]. Journal of Software, 2011, 22(2): 177−195(in Chinese)

(陈小红, 尹斌, 金芝. 基于问题框架方法的需求建模: 一个本体制导的方法 [J]. 软件学报, 2011, 22(2): 177−195)

[8] Elbendak M, Vickers P, Rossiter N. Parsed use case descriptions as a basis for object-oriented class model generation[J]. Journal of Systems and Software, 2011, 84(7): 1209−1223

[9] Pudlitz F, Brokhausen F, Vogelsang A. Extraction of system states from natural language requirements[C]//Proc of the 27th IEEE Int Requirements Engineering Conf (RE). Piscataway, NJ: IEEE, 2019: 211-222

[10] Qian Chen, Wen Lijie, Kumar A, et al. An approach for process model extraction by multi-grained text classification[C]//Proc of the 32nd Conf on Advanced Information Systems Engineering(CAiSE). Berlin: Springer, 2020: 268-282

[11] OpenAI. ChatGPT. [2023-08-01] .https://openai.com/blog/chatgpt/. 2023.09

[12] Dong Qingxiu, Li Lei, Dai Damai, et al. A survey for in-context learning[J]. arXiv preprint, arXiv: 2301. 00234, 2022

[13] Li Ge, Peng Xin, Wang Qianxiang, et al. Challenges from LLMs as a Natural Language Based Human-machine Collaborative Tool for Software Development and Evolution[J]. Journal of Software, 2023, 34(10): 4601−4606(in Chinese)
(李戈, 彭鑫, 王千祥, 等. 大模型: 基于自然交互的人机协同软件开发与演化工具带来的挑战 [J]. 软件学报, 2023, 34(10): 4601−4606)

[14] Grady J O. System requirements analysis[M]. Cambridge: Elsevier, 2010

[15] Katzenbach J R, Smith D K. The wisdom of teams: Creating the high-performance organization[M]. Boston: Harvard Business Review Press, 2015

[16] Belbin R M, Brown V. Team roles at work[M]. Oxfordshire: Routledge, 2022

[17] Jackson M. Problem frames: analysing and structuring software development problems[M]. Boston: Addison-Wesley, 2001

[18] Jin Dongming, Wang Chunhui, Jin Zhi. Automating extraction of problem diagrams from natural language requirements document[C]//Proc of the 31st IEEE Int Requirements Engineering Conference Workshops (REW). Piscataway, NJ: IEEE, 2023

[19] Chen Xiaohong, Jin Zhi. An ontology-guided process for developing problem frame specification: an example[C]//Proc of the 3rd Int Workshop on Applications and Advances of Problem Frames(IWAAPF'08) . Leipzig: ACM, 2008: 36-39

[20] Xing Zhenchang, Huang Qing, Cheng Yu, et al. Prompt sapper: LLM-empowered software engineering infrastructure for AI-native services[J]. arXiv preprint, arXiv: 2306.02230, 2023

[21] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in Neural Information Processing Systems, 2017, 5998-6008

[22] Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint, arXiv: 1810. 04805, 2018

[23] Brown T, Mann B, Ryder N, et al. Language models are few-shot learners[J]. Advances in Neural Information Processing Systems, 2020, 33: 1877−1901

[24] Bajaj D, Goel A, Gupta S C, et al. MUCE: A multilingual use case model extractor using GPT-3[J]. International Journal of Information Technology, 2022, 14(3): 1543−1554.

[25] Leong I T, Barbosa R. Translating Natural Language Requirements to Formal Specifications: A Study on GPT and Symbolic NLP[C]//Proc of the 53rd Annual IEEE/IFIP Int Conf on Dependable Systems and Networks Workshops (DSN-W). Piscataway, NJ: IEEE, 2023: 259-262

[26] Chen Boqi, Chen Kua, Hassani S, et al. On the Use of GPT-4 for Creating Goal Models: An Exploratory Study[C]//Proc of MoDRE workshop at Requirement Engineering conf. Hannover. Piscataway, NJ: IEEE, 2023: 262-271

[27] Liu Pengfei, Yuan Weizhe, Fu Jialan, et al. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing[J]. ACM Computing Surveys, 2023, 55(9): 1−35

[28] Wei Jason, Wang Xuezhi, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models[J]. Advances in Neural Information Processing Systems, 2022, 35: 24824−24837

[29] Ross D T, Schoman K E. Structured analysis for requirements definition[J]. IEEE Transactions on Software Engineering, 1977(1): 6−15

[30] Graham I. Requirements engineering and rapid development: an object-oriented approach[M]. Boston: Addison-Wesley Press, 1998

[31] Vinay S, Aithal S, Desai P. An approach towards automation of requirements analysis[C]//Proc of the Int Multi Conf of Engineers and Computer Scientists. Piscataway, NJ: IEEE, 2009: 1

[32] Umber A, Bajwa I S, Asif Naeem M. NL-based automated software requirements elicitation and specification[C]//Proc of 1st Conf on Advances in Computing and Communications (ACC). Berlin: Springer, 2011: 30-39

[33] Bajwa I S, Choudhary M A. From natural language software specifications to UML class models[C]//Proc of 13th Int Conf on Enterprise Information Systems(ICEIS). Berlin: Springer, 2012: 224-237

[34] Bajwa I S, Asif Naeem M. On specifying requirements using a semantically controlled representation[C]//Proc of 16th Int Conf on Applications of Natural Language to Information Systems (NLDB). Berlin: Springer, 2011: 217-220

[35] Umber A, Bajwa I S. A step towards ambiguity less natural language software requirements specifications[J]. International Journal of Web Applications, 2012, 4(1): 12−21

[36] Pohl K, Böckle G, van der Linden F, et al. Overview of the example domain: Home automation[J]. Software Product Line Engineering: Foundations, Principles, and Techniques, 2005: 39-52

[37] Jin Zhi, Liu Lin, Chen Xiaohong, et al. Software Requirements Engineering Methods and Practices[M]. Beijing: Tsinghua University Press, 2023(in Chinese)
(金芝, 刘璘, 陈小红, 等. 软件需求工程方法与实践 [M]. 北京: 清华大学出版社, 2023)

[38] Al-Hroob A, Imam A T, Al-Heisa R. The use of artificial neural networks for extracting actions and actors from requirements document[J]. Information and Software Technology, 2018, 101: 1−15
[39] LangchainAI. LangChain. [2023-08-01]. https://www.langchain.com/
[40] THUDM. ChatGLM. [2023-08-01].https://github.com/THUDM/ChatGLM-6B
[41] Yang Mengfei, Gu Bin, Duan Zhenhua, et al. Intelligent program synthesis framework and key scientific problems for embedded software[J]. Chinese Space Science and Technology, 2022, 42(4): 1−7(in Chinese)
(杨孟飞, 顾斌, 段振华, 等. 嵌入式软件智能合成框架及关键科学问题 [J]. 中国空间科学技术, 2022, 42(4): 1−7)
[42] Gu Bin, Dong Yunwei, Wang Zheng. Formal modeling approach for aerospace embedded software[J]. Journal of Software, 2015, 26(2): 321−331(in Chinese)
(顾斌, 董云卫, 王政. 面向航天嵌入式软件的形式化建模方法 [J]. 软件学报, 2015, 26(2): 321−331)



Jin Dongming, born in 2001. PhD candidate. His main research interest includes intelligent requirements engineering.
靳东明, 2001 年生. 博士研究生. 主要研究方向为智能化需求工程.



Jin Zhi, born in 1962. PhD, professor. Her main research interests include requirements engineering and software automation.
金　芝, 1962 年生. 博士, 教授. 主要研究方向为需求工程、软件自动化.



Chen Xiaohong, born in 1982. PhD, associate professor. Her main research interests include requirements engineering, and knowledge based software engineering and formal method.
陈小红, 1982 年生. 博士, 副教授. 主要研究方向为需求工程、基于知识的软件工程和形式化方法.



Wang Chunhui, born in 1979. PhD, associate professor. Her main research interests include software engineering and requirements engineering.
王春晖, 1979 年生. 博士, 副教授. 主要研究方向为软件工程、需求工程.